GRAPH BASED SEMI-SUPERVISED LEARNING IN COMPUTER VISION

by

NING HUANG

A dissertation submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

in conjunction with

The Graduate School of Biomedical Sciences

the University of Medicine and Dentistry of New Jersey

in partial fulfillment of the requirements

for the Joint Degree of

Doctor of Philosophy

Graduate Program in Biomedical Engineering

Written under the direction of

Joseph Wilder

and approved by

_____

_____

_____

_____

New Brunswick, New Jersey

January, 2009

ABSTRACT OF THE DISSERTATION

Graph Based Semi-Supervised Learning in Computer Vision

by Ning Huang

Dissertation Director: Joseph Wilder

Machine learning from previous examples or knowledge is a key element in many image processing and pattern recognition tasks, e.g. clustering, segmentation, stereo matching, optical flow, tracking and object recognition. Acquiring that knowledge frequently requires human labeling of large data sets, which can be difficult and time-consuming to obtain. One way to ameliorate this task is to use Semi-supervised Learning (SSL), which combines both labeled and raw data and incorporates both global consistency (points in the same cluster are likely to have the same label) and local smoothness (nearby points are likely to have the same label). There are a number of vision tasks that can be solved efficiently and accurately using SSL. SSL has been applied extensively in clustering and image segmentation. In this dissertation, we will show that it is also suitable for stereo matching, optical flow and tracking problems.

Our novel algorithm has converted the stereo matching problem into a multi-label semi-supervised learning one. It is similar to a diffusion process, and we will show our approach has a closed-form solution for the multi-label problem. It sparks a new direction from the traditional energy minimization approach, such as Graph Cut or Belief Propagation. The occlusion area is detected using the matching confidence level, and solved with local fitting. Our results have been applied in the Middlebury Stereo

database, and are within the top 20 best results in terms of accuracy and is considerably faster than the competing approaches.

We have also adapted our algorithm, and demonstrated its performance on optical flow problems. Again, our results are compared with the ground truth and state of the art on the Middlebury Flow database, and its advantages in accuracy as well as speed are demonstrated.

The above algorithm is also being used in our current NSF sponsored project, an Automated, Real-Time Identification and Monitoring Instrument for Reef Fish Communities, whose goal is to track and recognize tropical fish, initially in an aquarium and ultimately on a coral reef. Our approach, which combines background subtraction and optical flow, automatically finds the correct outline of multiple fish species in the field of view, and tracks the contour reliably over consecutive frames. Currently, near real-time results are being achieved, with a processing frame rate of 3-5 fps.

The recent progress in semi-supervised learning applied to image segmentation is also briefly reviewed.

Acknowledgements

First of all, I would really like to thank my advisor, Prof. Joseph Wilder for supervising my dissertation and endless help. I can not have finished this degree without his continuous help and support.

Eight years have passed since I began my graduate study here at Rutgers. I would like to thank the many friends who have made my life and my study here more enjoyable. Among others, Jingsheng, Lixia, Hongjun, Zhen, Alex, Xifan, Min, Ruoheng, Qi, Li, Sekhar, Kevin, Brian, Xiang, Hui, Kai, Ke, Lin, Danxi, Cathy, Hang, Jun, Jian, Beizhong.

Moreover, I like to thank Shen, Hong who has brought me in his group in Siemens as an intern to gain industrial experience as well as more insight into medical imaging and computer vision in general.

Dedication

To my parents, my sister and Daisy,

without whom I would not have been able to get to where I am today

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Machine learning from previous examples or knowledge is a key element in many image processing and pattern recognition tasks, e.g. clustering, segmentation, stereo matching, optical flow, tracking and object recognition. Acquiring that knowledge frequently requires human labeling of large data sets, which can be difficult and time-consuming to obtain. One way to ameliorate this task is to use Semi-supervised Learning (SSL), which combines both labeled and raw data and incorporates both global consistency (points in the same cluster are likely to have the same label) and local smoothness (nearby points are likely to have the same label).

There are a number of vision tasks that can be solved efficiently and accurately using SSL, such as image segmentation and object recognition. In this dissertation, we will show that it is also suitable for stereo matching, optical flow and tracking problems.

Our novel algorithm has converted the stereo matching problem into a multi-label semi-supervised learning one. It is similar to a diffusion process, and we will show our approach has a closed-form solution for the multi-label problem. It sparks a new direction from the traditional energy minimization approaches, such as Graph Cut or Belief Propagation. The occlusion area is detected using the matching confidence level, and solved with local fitting. Our results have been applied in the Middlebury Stereo database, and are within the top 20 best results in terms of accuracy and is considerably faster than the competing approaches.

We have also adapted our algorithm, and demonstrated its performance on optical flow problems. Again, our results are compared with the ground truth and state of the

art on the Middlebury Flow database, and its advantages in accuracy as well as speed are demonstrated.

The above algorithm will also be used in our current NSF sponsored project, an Automated, Real-Time Identification and Monitoring Instrument for Reef Fish Communities, whose goal is to track and recognize tropical fish, initially in an aquarium and ultimately on a coral reef. Our approach, which combines background subtraction and optical flow, automatically finds the correct outline of multiple fish species in the field of view, and tracks the contour reliably over consecutive frames. Currently, near real-time results are being achieved, with a processing frame rate of 3-5 fps. The recent progress in semi-supervised learning applied to image segmentation is also briefly reviewed.

1.2 A Brief Review of Semi-Supervised Learning

In recent years, machine learning algorithms, especially semi-supervised learning algorithms, have flourished in data clustering, graph partitioning and nonlinear dimension reduction as well as computer vision.

Learning can be classified roughly, into unsupervised learning, supervised learning and semi-supervised learning.

In **supervised learning**, the training instances are provided with correct labels, which give feedback about the learning result. It is applied extensively in speech recognition, hand-digit recognition and many other applications where training can provide significant improvements in performance.

In **unsupervised learning**, no labels or classifications are provided, and the problem becomes much harder since no prior examples are available from which to learn. Thus, it gives less accuracy than supervised learning.

Labeled points are more expensive to get than unlabeled points for various reasons. Most importantly, labeled points need human or even experts' annotations or interactions, which is time-consuming. Meanwhile unlabeled data may be relatively easy to collect, but they do not give feedback information about the leaning results and are seldom used in training.

Semi-supervised learning (SSL) addresses this dilemma by using a small amount of labeled data, together with largely unlabeled data, for better learning. Because semi-supervised learning requires less prior knowledge about data and gives competitive accuracy, it is of great interest both in theory and in practice. Knowledge can be propagated from labeled to unlabeled points, given the neighborhood structure of the data. Thus, limited labeled data can be enough to get good learning results.

In mathematical terms, assume there are n data points, with l labeled points $X_l := x_1, x_2, ..., x_l$, and $u = n - l$ unlabeled points $X_u := x_{l+1}, ..., x_n$, typically $l << u$. And $Y := y_i \ i = 1...l$ are the labels of the l labeled points x_i. We are interested to infer the learning function f which best predicts $f(x) = y \ for \ all \ x \in X$.

In some cases, Semi-Supervised Learning will yield better results than Supervised learning, which requires the unlabeled data X_u carry enough information to infer the unknown labels Y_u. To generalize a finite training set to a much larger test cases, two underlying assumptions are made normally, [30] [4] .

The first is the **Local Smoothness** assumption, if two points x_1 and x_2 are close and lie in a high-density region, then the corresponding outputs y_1 and y_2 should be similar as well. Or in short, nearby points are likely to have the same label.

The second one is the **Global Consistency** assumption, if x_1 and x_2 are in the same cluster or on the same manifold, they are likely to have the same labels y. Note: points on the same cluster are not necessarily close. The second assumption could also be stated in another way as, the decision boundary between clusters should lie in a low-density region.

An alternative assumption which has used in several SSL, manifold learning and dimension reduction methods, is the **Manifold Assumption**, which states,

The high dimensional data lie on a low-dimensional manifold.

There are many semi-supervised learning methods, some frequently used methods include: Expectation Maximization (EM), self-training, co-training, transductive support vector machines and graph-based methods. A survey on Semi-supervised learning could be found on [31] .

For computer vision applications, we are focusing on graph-based methods. The

graph is composed data nodes, either labeled or unlabeled, and connected by edges. Nodes connected by heavy-weighted edges are likely to have the same label according the local smoothness assumption.

Formally, a connected graph $G = (V, W)$ can be built with vertex V corresponding to the n data points, and W the edge weight matrix. The $n * n$ symmetric definite weight matrix W is provided, usually using a Gaussian kernel

$$W_{ij} = exp(-\frac{||I_i - I_j||^2}{2\sigma^2}) \tag{1.1}$$

where i and j are adjacent nodes.

The learning function f can be built by relaxing discrete labels to continuous values, which satisfies

$$f(x_i) = y_i \ for \ i = 1...l \tag{1.2}$$

To satisfy the local smoothness and global consistency assumptions, we seek to minimize the energy

$$E = \sum_{i \in N_j} w_{ij}(f(x_i) - f(x_j))^2 \tag{1.3}$$

It is closely related to the graph Laplacian Δ,

$$\Delta = D - W$$

Sometimes the normalized Laplacian is used interchangeably

$$L = D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}}$$

where W is the weight matrix, and D is the diagonal degree matrix,

$$D_{ii} = \sum_{j=1}^{n} W_{ij}$$

And the energy could be rewritten as

$$E = \sum_{i \in N_j} w_{ij}(f(x_i) - f(x_j))^2 = f^t \Delta f \tag{1.4}$$

still under constraint of Equation (1.2). So the non-smoothness along the edges of a weighted graph are penalized, the more the edge weight, the higher the penalty.

Let $f = [f_l \ f_u]^t$, where f_l denotes the values on labeled points, and f_u denotes the values on the unlabeled points, and divide the graph Laplacian into 4 blocks after the lth row and column, according to labeled and unlabeled points, where Δ_{ll} is the graph Laplacian within labeled points, Δ_{uu} is the graph Laplacian within unlabeled points, and $\Delta_{lu} = (\Delta_{ul})^T$ are the graph Laplacian between labeled and unlabeled points.

$$\Delta = \begin{bmatrix} \Delta_{ll} & \Delta lu \\ \Delta_{ul} & \Delta uu \end{bmatrix} \tag{1.5}$$

the solution is given by Zhu[8]

$$f_u = -\Delta_{uu}^{-1} \Delta_{ul} Y_l \tag{1.6}$$

where $Y_l = [y_i], \ i = 1, ..., l$, are the labels of the labeled points, (interested readers can refer to [32] for details of proof)

In Chapter 2, we will show see the above equations can be applied directly in image segmentation, and give useful results.

Semi-supervised learning is also related to many other areas, including among others, manifold learning, nonlinear dimension reduction, community structure.

1.3 Semi-Supervised Learning in Computer Vision

Grady [10] has proposed a random walk algorithm, which is a general interactive image segmentation algorithm. The supervised labels (or seed) are provided by the users. The probabilities are computed analytically using a sparse, symmetric, semi-definite set of linear equations. The approach is related to the graph Laplacian matrix Δ.

In Grady's follow-on work [9], he also develops the solution in the same problem, but with real-valued node-wise prior probabilities.

In [6], image segmentation with user-supplied seeds is viewed as a transduction problem. Segmentation is modeled as the task of finding the matting coefficients for

unlabeled points given matting coefficients on labeled points. It is based on the Laplacian graph regularization, and is thus essentially a semi-supervised learning method.

1.4 Organization and Contributions

The layout of this dissertation is as follows:

Chapter 2 gives an overview of Semi-supervised learning applied to image segmentation.

In Chapter 3, we give our semi-supervised learning framework, and apply it to dense stereo matching. Additional stereo matching postprocessing techniques are presented, which have been adapted to our SSL framework and data types. The results are presented at each step, and evaluated with the Middlebury Stereo database [24] and compared with the state of art.

In chapter 4, semi-supervised learning is applied in optical flow problems. The algorithm has been adapted to the spatial-temporal problem. Sparse feature matching algorithms have been combined into the framework, to solve the possible large flow fields. And a hierarchical approach is proposed to solve the problem efficiently. Again, the results are evaluated using the Middlebury Flow Database [1].

In chapter 5, an heuristic algorithm to solve the general MRF problem is proposed. It gives a boost to the computation of traditionally computationally expensive MRF problem and it is applied to real time tracking problems.

In chapter 6, I will present my conclusion, and discuss the advantages of my algorithms, especially the computational efficiency. Possible directions of future work are also outlined.

Chapter 2

Graph-Based Semi-Supervised Learning in Image Segmentation

The graph-based semi-supvervised learning and related spectral methods view the image segmentation problem as a clustering algorithm. Each segment in an image corresponds to a cluster in a point cloud.

Recall the graph construction and Graph Laplacian Δ introduced in the last chapter.

A connected graph $G = (V, W)$ can be built with vertex V corresponding to the n data points, and W the edge weight matrix. The $n * n$ symmetric definite weight matrix W is provided, usually using a Gaussian kernel

$$W_{ij} = exp(-\frac{||I_i - I_j||^2}{2\sigma^2}) \tag{2.1}$$

The graph Laplacian Δ is given by,

$$\Delta = D - W$$

where W is the affinity weight matrix, and D is the diagonal degree matrix,

$$D_{ii} = \sum_{j=1}^{n} W_{ij}$$

W as well as Δ encodes the pairwise relationship between all pairs of pixels, and it is not so obvious how we can use that relationship directly to infer the clustering of pixels, so as to accomplish image segmentation.

One way to do that is to use spectral methods, which utilize the eigenvalue decomposition of the Graph Laplacian. A milestone in this direction is the normalized cut work proposed by Shi and Malik [26].

The normalized cut (NCut) is defined as

$$N_{cut}(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)} \tag{2.2}$$

where $cut(A,B)$ is the total dissimilarity between two parts A and B, which is the sum of edge weights from A to B

$$cut(A,B) = \sum_{u \in A, v \in B} w(u,v) \tag{2.3}$$

$assoc(A,V)$ is the total connections from nodes in A to all nodes in the graph V. NCut computes the total costs as a fraction of the total edge connections to all the nodes in the graph.

In minimizing the NCut, it converts into solving an eigenvalue problem, and is formulated to use the second smallest eigenvector to bipartition the graph.

In Ng et al's work [22], the normalized framework has been expanded into general spectral space, and instead of bipartitioning the graph, the points are clustered into k clusters. The steps are as below,

(1) build the affinity weight matrix W

(2) compute degree matrix D, and the normalized Laplacian matrix $L = D^{\frac{1}{2}}WD^{\frac{1}{2}t}$ (The Laplacian matrix and the normalized Laplacian are similar matrices, and they have the same eigenvectors)

(3) Find the k largest eigenvectors $v_1, v_2, ..., v_k$of L, orthogonal to each other, and stack them column-wise to form the matrix V. (Optionally, you can normalize X row-wise, i.e., make row sum to 1)

(4) Now essentially we have converted the original clustering problem into a clustering problem in its spectral space. Treat each row of X as a point in spectral space, they can be grouped into k clusters using K-means or any other algorithm.

The spectral space is a lower dimensional manifold compared with the original space to cluster. According to the Manifold Assumption in Chapter One, "The high dimensional data lie on a low-dimensional manifold", it is more insightful to cluster data in the low manifold space, where they form tighter clusters which yield better clustering results.

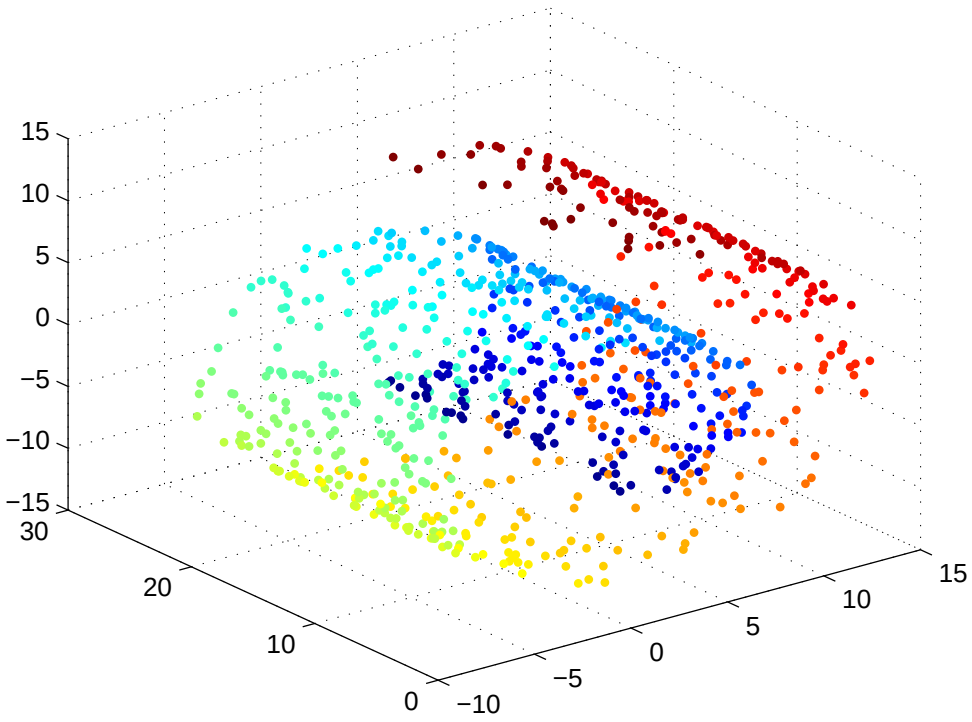

Figure 2.1: 3D data points lie on a warped 2D surface, the so called Swiss Roll, an example of how the high dimensional data actually lie on a low dimensional space. It can be thought of as a 2D plane warped into a 3D roll, thus it is a 2D manifold in a 3D space.

An typical example is in 2.1, which shows the high dimensional data actually lie on a low dimensional space, in this case, a swiss-roll like warped surface.

Both normalized cut [26] and spectral clustering [22] make assumptions on the number of clusters. Normalized cut assumes binary cut, and in spectral clustering, the cluster number k is given in advance (So does K-means). [27] is another spectral method which studies the eigenvalues and tries to determine the best cluster number automatically, by comparing the successive ratios of eigenvalues. Edge separators of a graph are produced by iteratively reweighting the edges until the graph disconnects into the prescribed number of clusters.

All those methods try to convert the original clustering algorithm into a clustering problem in a low dimensional space, i.e., spectral space. Normally, eigenvector decomposition methods are used. Thus they are, in general, spectral clustering algorithms.

Other algorithms try to divide the square edge weight affinity matrix into blocks. The unknown number of clusters and lack of knowledge of either cluster centers or must-links and cannot-links make the problem hard to get satisfactory results.

On the other hand, Graph-Based semi-supervised learning methods generally provide some labels via user interactions. The edge weight affinity matrix could be viewed as the transition probability matrix from one pixel to the other. And the image segmentation problem can be formulated as a random walk problem. Each point is free to walk to neighboring points with the probability of the edge weights. Each point is assigned the same label as the most likely seed it is going to reach.

Grady [10] pioneered in using the Semisupervised learning framework in image segmentation, where the seed points are supervised labels. Again, the connected graph $G = (V, W)$ is built with vertex V corresponding to the n data points, and W the edge weight matrix. A set of labeled nodes is provided for each segment.

Independently, Grady finds a similar solution to the Semisupervised learning problem in Zhu et al's work [33],

First the graph Laplacian is decomposed into labeled and unlabeled parts,

$$\Delta = \begin{bmatrix} \Delta_{ll} & \Delta lu \\ \Delta_{ul} & \Delta uu \end{bmatrix} \tag{2.4}$$

$$f_u = -\Delta_{uu}^{-1} \Delta_{ul} Y_l$$

In Zhu's work, $Y_l = [y_i],\ i = 1, ..., l$, are the binary labels of the labeled points, e.g. $y_i \in 0, 1$. To make it suitable for a multilabel segmentation problem, Grady makes Y_l is a $l * k$ matrix, where k is the number of all possible labels, and

$$Y_{ij} = \begin{cases} 1 & \text{if } y_i = j \\ 0 & \text{if } y_i \neq j \end{cases}$$

The figure 2.2 below shows an image segmented using random walk with seed points provided by the user.

Figure 2.2: Image segmentation via user interaction. On the left, image of a kid, with white and black dots as seeds points for foreground and background. On the right, image segmentation results using random walk.

Graph cut [3] is similar in viewing the whole image as a graph, and using graph algorithms, especially max-flow, min-cut algorithms to find the best graph partition of the image. But graph cut computes the minimum cut that separates two regions, and for multi-label problems, it has to convert into sequential binary cut problems.

Recently, Duchenne and Audibert reformulated the problem and gave more insight into the semi-supervised learning applied to image segmentation in [6].

All approaches need user interaction to provide the labeled nodes for different segments. So it is truly semi-supervised for image segmentation problems. This is in direct comparison with our approach in stereo matching and optical flow in the following chapters. Although they are also based on the semi-supervised learning framework, no user interaction is ever needed.

Chapter 3

Stereo Matching Using Semi-Supervised Learning

A novel algorithm for stereo matching and optical flow problems is proposed, which uses a multi-label semi-supervised learning approach. It reformulates the common energy minimization problem into a diffusion process, which we show has a closed-form solution. With global consistency and local smoothness in disparity, disparity field or optic flow can be solved with less computation.

Experimental results demonstrate the efficiency, robustness and accuracy of this algorithm.

The structure of this chapter is as follows:

In Section 3.1, we will give the formal formulation of the stereo problem. In Section 3.2, the diffusion algorithm will be presented with a closed-form solution. In Section 3.3, 3.4 and 3.5, some post-processing technique will be introduced to further improve our results. Section 3.3 integrates global knowledge using image segmentation information. Section 3.4 introduces noise detection. Section 3.5 presents noise removal and outlier handling by local information. Section 3.6 summarizes the framework. Finally, section 3.7 demonstrates the results and gives a conclusion.

A high level sketch of the overall framework is shown in Figure 3.1,

The details are presented in the various sections outlined above.

3.1 Problem Formulation

Assume we are given two input images I and I', which could be left and right images in stereo or two consecutive frames in optic flow and tracking.

The traditional Markov Random Field (MRF) approach tries to find optic flow or disparity fields on image I, by energy minimization. The total energy to minimize in

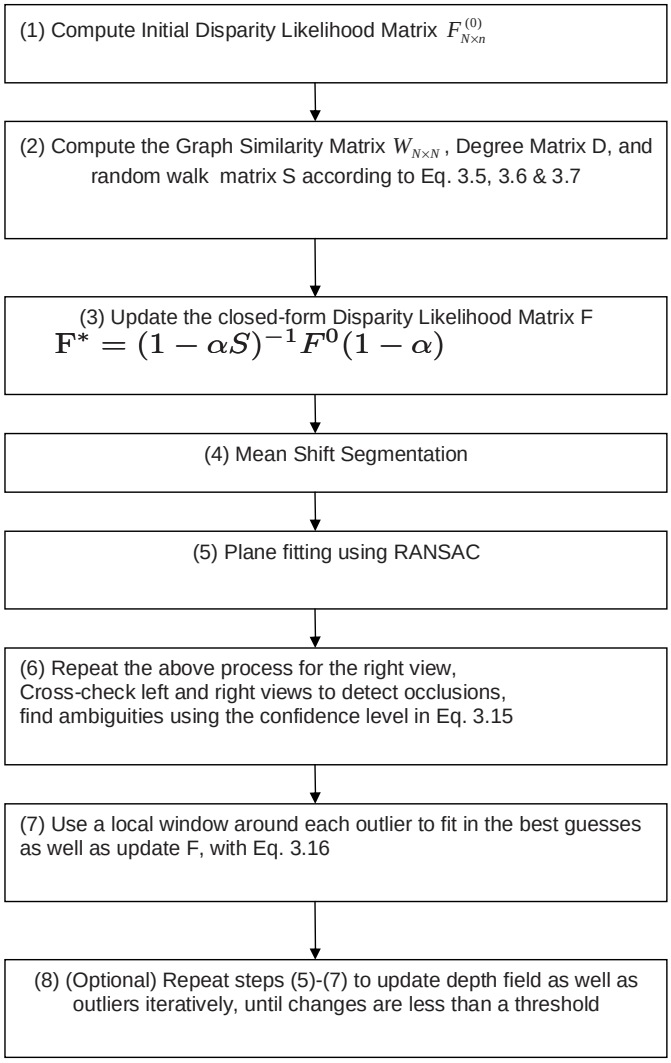

Figure 3.1: The high-level flow chart of the proposed approach on stereo matching problem.

MRF is as follows:

$$E(f) = \sum_{p} D_p(f_p) + \sum_{p,q \in N} V_{pq}(f_p, f_q) \tag{3.1}$$

Where f_p is the flow vector at node p, $D_p(f_p)$ is the data term, and $V_{pq}(f_p, f_q)$ is the smoothness term, between neighboring nodes p and q. $E(f)$ is composed of the sum of all data terms and smoothness terms.

A different, novel approach is proposed to convert the energy minimization problem in MRF into a semi-supervised Maximum A Posteriori (MAP) problem, by making the posterior probability maximum.

Consider all of the pixels in image I as a point set $X = x_1, x_2, ..., x_N$, and make all of the possible disparities between I and I', as a label set $L = 1, 2, ..., n$, where N is the total number of pixels, and n is total number of disparity labels.

An initial disparity likelihood matrix $F0$, for pixel $i \in X$, with a depth value $d \in L$ is given by the inverse exponential of the Sum of Squared Distances (SSD)

$$F0(i,d) = exp(-\frac{\sum_{j \in N_i} w_{ij} |I(j) - I'(j+d)|^2}{2\sigma^2 \sum_{j \in N_i} w_{ij}}) \tag{3.2}$$

where SSD is computed in the neighborhood of point i including i itself, j are the neighbors of i, which could be 4-neighborhood, or k Nearest Neighbors (kNN). w_{ij} is the edge weight between i and j. This form works like an adaptive support, which assigns strong weights on similar pixels with i in the local neighborhood. The inverse exponential converts the SSD into a similarity measure as well as regulates the disparity field F between 0 and 1. σ is a regularization factor. $F0$ is a matrix of size $N * n$ computed on all points in X, with every possible label $d \in L$.

The disparity likelihood matrix $F0$ is then converted into a soft assigned probability matrix by normalization

$$F^0(i,d) = \frac{F0(i,d)}{\sum_d F0(i,d)} \tag{3.3}$$

where each element $F^0(i,d)$ represents the initial probability of point i to be given a disparity vector d. F^0 is then a non-negative matrix and its row sum is 1.0. Intuitively,

the bigger $F^0(i,d)$ of a given point i, the higher the probability that point i is assigned with disparity d. The maximum likelihood disparity field R could be computed as

$$R(i) = \max_{d \in L} F^0(i,d) \tag{3.4}$$

for any $i \in X$. Of course lots of noise and errors could be expected as shown in Figure 3.2 (d).

3.2 Diffusion Process and Closed-Form Solution

In what follows, we will show how a diffusion process could solve the problem of noise and errors. The local consistency assumption is made here, where the disparity likelihood matrix $F(i)$ at a given point i should be affected by all its neighbor points $j \in N_i$.

3.2.1 Diffusion Process

An edge weight matrix W, which measures the similarities between neighboring nodes, is constructed by

$$W_{ij} = \begin{cases} exp(-\frac{|I_i - I_j|^2}{2\sigma^2}) & i,\ j \in N \\ 0 & otherwise \end{cases} \tag{3.5}$$

And the random walk matrix is given by S, which is the weight matrix W normalized by its column sums D. S could be thought of as a probabilistic transition matrix, with each element S_{ij} indicating the transition probabilities from node i to j.

$$S = D^{-1} \cdot W \tag{3.6}$$

where D is the degree matrix (column sum) of W

$$D_{ik} = \begin{cases} \sum_j W_{ij} & \text{if } i = k \\ 0 & \text{if } i \neq k \end{cases} \tag{3.7}$$

The diffusion process is given by

$$F^{t+1} = \alpha S F^t + (1-\alpha) F^0 \tag{3.8}$$

where F^{t+1} is the next disparity field and F^t is the current disparity field. $\alpha \in [0, 1]$ is a weighting coefficient. The first term gives a one-step diffusion (or random walk) from the current disparity, while the 2nd term accounts for that the new disparity field should be similar with the initial disparity field F^0.

The underlining reason to use the sparse $n * n$ matrix S is that the label of a given point i is affected by all its neighbors $j \in N_i$, the stronger their edge weights w_{ij}, the more similar should be label at i with the label at j. Coefficient $(1-\alpha)$ determines how similar the final solution is to the initial disparity field F^0. 0.95 is chosen for α in our experiments, which gives the best empirical results.

3.2.2 Derivation of the Closed-Form Solution

Below we will show that the series, F^t converges.

From Equation (6.1.1)

$$F^1 = \alpha S F^0 + (1-\alpha) F^0$$

$$F^2 = \alpha S(\alpha S F^0 + (1-\alpha) F^0) + (1-\alpha) F^0$$

by substituting F iteratively, we get

$$F^t = (\alpha S)^t F^0 + (1-\alpha) \sum_{i=0}^{t-1} (\alpha S)^i F^0$$

since $0 < \alpha < 1$ and eigenvalues of S are between -1 and 1

$$\lim_{t\to\infty} (\alpha S)^t = 0, \ \lim_{t\to\infty} \sum_{i=0}^{t-1} (\alpha S)^t = (1-\alpha S)^{-1} \tag{3.9}$$

hence

$$F^* = \lim_{t\to\infty} F^t = (1-\alpha)(1-\alpha S)^{-1} F^0$$

or simply

$$F^* = (1 - \alpha S)^{-1} F^0 (1 - \alpha) \tag{3.10}$$

The Max A Posterior estimate of the disparity D after the diffusion process is

$$d_i = \arg\max_d F(i, d), \tag{3.11}$$

and the confidence on the given pixel i.

$$C_i = max_d F(i, d) \tag{3.12}$$

The diffusion end result F, has taken care of the matching costs of neighboring pixels for every individual pixel, and drastically improves over the initial matching cost volume F^0, which is shown in Figure 3.2.

3.2.3 Alternative Derivation

From a regularized energy minimization point of view, we are trying to minimize the following cost function

$$E(F) = E_l(F) + \mu E_s(F)$$

where

$$E_l(F) = \sum_{i \in L} (f_i - y_i)^2$$

is the fitting term, which poses the constraints that a good classifying function should not change too much from the initial label assignment.

And

$$E_s(F) = \sum_{i \in N_j} w_{ij} (f(x_i) - f(x_j))^2$$

is the smoothness term, μ is a weighting coefficient.

Differentiating $E(F)$ with respect to F, we get

$$\frac{\partial E}{\partial F} = F - SF + \mu(F - F^0) = 0$$

and could be transformed into

$$((1+\mu)I - S)F = \mu Y$$

or equivalently

$$(I - \frac{1}{1+\mu}S)F = \frac{\mu}{1+\mu}Y$$

let $\alpha = \frac{1}{1+\mu}$, we get

$$(I - \alpha S)F = (1-\alpha)Y$$

and exactly the same equations as in (3.10) is derived.

3.3 Global Constraint by Segmentation

In this section, a global constraint is posed to better regularize the diffusion process, using image segmentation results. This is especially helpful in stereo matching, where similar or close disparities are expected for each individual object.

The mean shift algorithm, proposed by Comaniciu and Meer [5], are used for our image segmentation purposes, because it is an unsupervised algorithm with good results. Image I is over-segmented into M different segments using mean shift. Each segment S_k will be assigned a segment number k, where $k \in 1...M$. Each segment is likely to lie on the same surface with no discontinuity, either on a frontal-parallel plane, or on a piece-wise smooth surface.

An RANSAC (RANdom SAmple Consensus) [2] plane fitting approach has been taken to boost the results obtained in the previous section. It uses the assumption that each segment should lie on a piece-wise smooth plane.

3.3.1 Plane Fitting Approach

Points on a segment lie on the same surface without discontinuity. Most of the time, the surface is a piece-wise smooth plane. Once we have the initial disparities, we can use RANSAC to fit that plane.

RANSAC fits a plane by randomly selecting 3 points, then it checks how many observed points lies within a small distance to the plane (which are called inliers). It continues to pick the plane randomly until it finds the one which has the most number of inliers or it reaches a maximum number of iterations.

The plane fitting disparity D_{pf} is given by RANSAC Plane fitting on D, the disparity field after the diffusion process.

As we are having a disparity probability volume, F_{N*n}, we can increase the probability values associated or close to D_{pf}, and decrease the probability values far away from D_{pf}, as opposed to taking directly $D = D_{pf}$.

3.4 Outlier Detection

After the global boosting by mean shift segmentation, the results are still not perfect due to occlusion in the left and right view. Outliers exist due to occlusions, ambiguities in homogeneous area and sometimes bad segmentation.

3.4.1 Occlusion

The right disparity field from the right image to the left image could be obtained using the same approach, just switch I and I', and change the range of disparity. To address occlusion explicitly, the left and right disparity fields are cross checked. The error at a given pixel x is

$$Error(x) = |d_l(x) - d_r(x + d_l(x))| \tag{3.13}$$

where $d_l(x)$ and $d_r(x)$ are the disparities of the left and right view, respectively, x is a point in the left view, and $x + d_l(x)$ is its corresponding point in the right view. If there is no occlusion, there should be a unique match on left and right view. The disparity from left to right should be the same with the disparity from right to left and $Error(x) = 0$. On the other hand, if x is occluded in the right view, the match point found on the other view, $x + d_l(x)$ is incorrect, and it could not be matched back to the same point of x in general. Thus, a potential occlusion is detected whenever

$Error(x) > 0$.

3.4.2 Ambiguities

To find possible low confidence disparities and ambiguities, we checked the confidence values C (defined in Eqn. 3.12)in the left disparity fields and F_r in the right disparity field. And we define total confidence T at a given point x as

$$T(x) = C(x) * F_r(x + d_l(x), d_l(x)) \tag{3.14}$$

where $F_r(x + d_l(x), d_l(x))$ is the likelihood of the corresponding point of x in the right view, to match x. If a match is unique, both $C(x)$ and $F_r(x + d_l(x), d_l(x))$ are expected to have a big value, and their product is big as well. When ambiguities exists, our confidence of the match becomes low because both terms are small. It measures the distinctness of the match.

3.4.3 Outlier Detection

Combining the Occlusion term and the ambiguity term, the outliers are defined as,

$$U(x) = \begin{cases} 1 & \text{if } Error(x) > \delta_1 \text{ or } T(x) > \delta_2 \\ 0 & Otherwise \end{cases} \tag{3.15}$$

where δ_1 is set to 1 and δ_2 is set to 0.25.

3.5 Local Confidence Fitting

For those outliers, defined by $U(x) = 1$, the disparity fields between two views does not give satisfying results due to either occlusion or ambiguities. The local smoothness within the same view will be used instead.

A local window of size 33*33 is used, an outlier centered at the window should have consistent depth value as those both close in distance and similar in color (which indicates they come from the same part or object). So we use the sum of all inliers within that window, weighted by color and spatial proximity.

Mathematically,

$$F(i,*) = \sum_{j \in W_i} w_{ij} * F(j,*) \tag{3.16}$$

where $w_{ij} = exp(-\frac{|i-j| \cdot |I_i - I_j|}{\sigma_2^2})$, depending on both the spatial distances $|i-j|$ and color similarities $|I_i - I_j|$ within the window W_i, and * represents all possible disparity values.

Then the Maximum likelihood of F is used to retrieve the disparity field after outlier fitting. The result after local fitting is shown in Figure 3.2(f).

3.6 Iterative Updating of Disparities and Outliers

We achieve very good results using the steps above. In Tsukuba pairs, we get an initial error percentage of 1.71%. This result on disparity can be further used to update the outlier fitting, and the disparity could be cross-checked and updated again in an Expectation-Maximization framework. The error rate of Tsukuba pairs improves to 1.36% on the 2nd iteration, which is within the state of art on the Middlebury dataset citeMiddleburyStereo.

The final result of the disparity field after global constraint is given in Figure 3.2.

3.7 Results and Further Work

3.7.1 Results

Results on the Tsukuba pair, including all the intermediate steps explained in the previous sections are shown in Figure 3.2.

The other image pairs in the Middlebury stereo database are given in Figure 3.3.

The computation of the stereo pairs are very efficient. The closed-form solution only takes O(nN) computations, where N is the number of pixels, and n is the number of labels. S in (3.10) is an $N * N$ sparse matrix, and similar as a penta-diagonal matrix, which could be solved in O(N). The running time of our algorithm takes about 2 minutes.

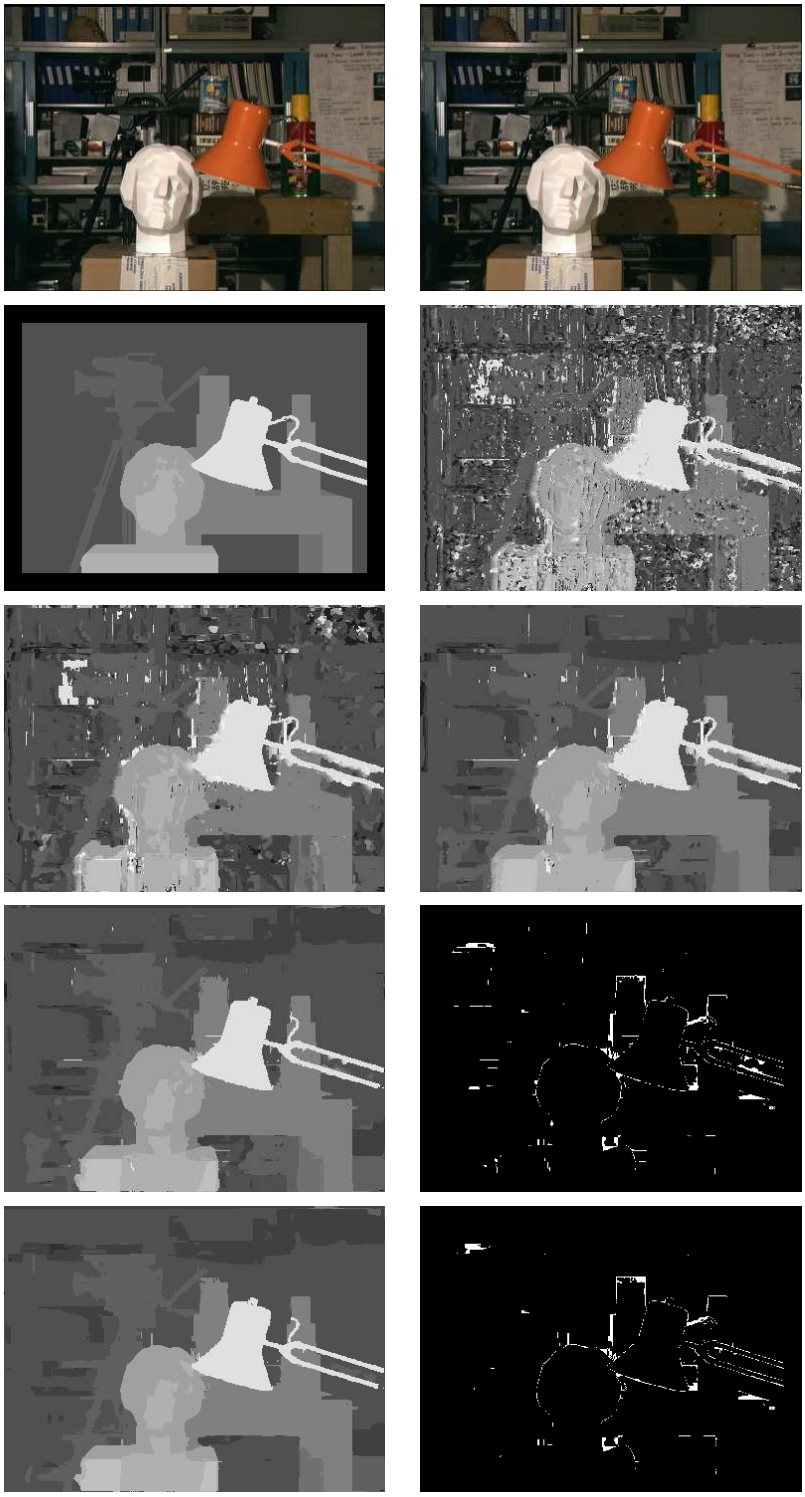

Figure 3.2: Results of the proposed algorithm on Tsubuka pair. (a) left image (b) right image (c) ground truth, (d) brute-force approach using MAP, (e) after the closed form solution of diffusion (f) after cross check and plane fitting (g) after first iteration (h) error of the first iteration (i) after second iteration (j) error of the second iteration

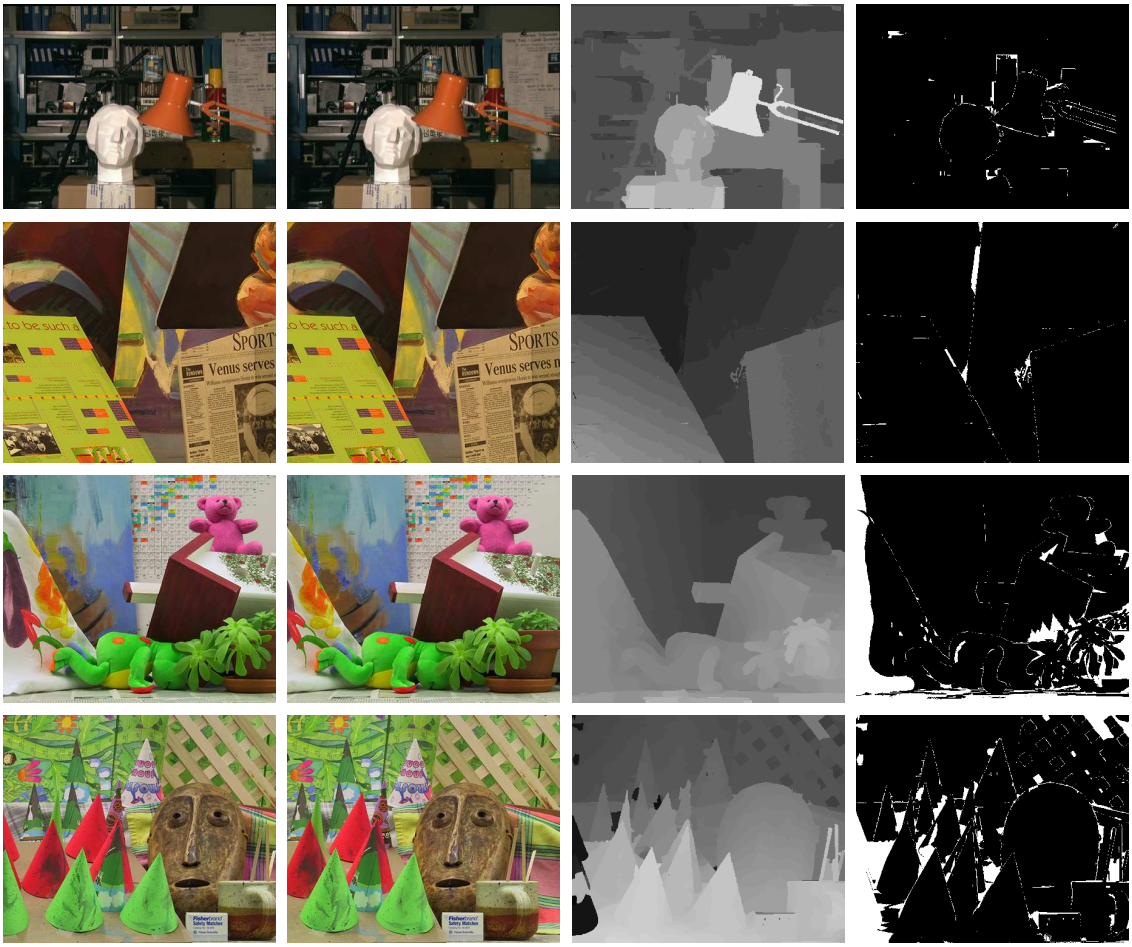

Figure 3.3: Results on all 4 pairs in Middlebury dataset, from left to right, left image, right image, results using our algorithm and the error maps. From top to bottom, Tsukuba, Venus, Teddy and Cone pairs. Error rates: 1.36, 0.67, 7.98, 5.76 respectively. Note: for Teddy and Cone pairs, the errors come mostly from the left-right view occlusion.

3.7.2 Future Work

To improve the results and integrate the post-processing implicitly into the closed-form solution, we may want to update the weight matrix W dynamically. For example, the mean shift segmentation results could be combined with the weight matrix computation, where edges e_{ij} across two different segments are assigned a much lower weight w_{ij}, than pixels on the same segment.

In the same manner, when occlusion map are estimated, the information can be put into the weight matrix computation again to improve the results.

Chapter 4

Optical Flow Using Semi-Supervised Learning

Optical Flow is one of the most challenging problems in computer vision. The motion vectors between consecutive frames are being estimated. It is an ill-posed problem because it is seriously under constrained. And, normally two common constraints are posed: constant brightness and spatial smoothness.

The major challenge of the optical flow problem is (a) illumination variation across different frames and (b) lack of information in untextured regions.

The figure below shows two consecutive frames of the "Army" sequence. The displacements between the frames vary according to the pixel location on the image. The Ground truth is given in 4.1(c), where different colors encode different flow directions and the saturation of color gives the magnitude of the flow at a given pixel. The codec for the flow encoding color can be read in a "color wheel" shown in 4.1(d).

4.1 Related Work

Lukas and Kanade [18], Horn and Schunck [11], have been the early pioneers on Optical Flow. Recent approaches include variational methods, MRF-based energy minimization approaches, learning and statistical techniques.

Lukas and Kanade [18] makes the brightness constancy constraint, and for each pixel (x, y) at time t,

$$I(x, y, t) = I(x + dx, y + dy, t + dt) \tag{4.1}$$

For the equation at each pixel, there are two unknowns. The ill-posed problem (the so-called aperture problem) is solved by the additional assumption that the flow (dx, dy) is constant within a small window.

Figure 4.1: Army Sequence, (a) frame 10 (b) frame 11 (c) Ground truth flow encoded in (d)Color Wheel, which different colors represent different flow orientations, and saturations represent flow magnitude.

Horn and Schunck formulate the optical flow problem into a variational problem [11] A global energy function f is sought to be minimized,

$$f = \int (\nabla I \cdot V + I_t)^2 + \alpha(|V_x|^2 + |V_y|^2)dxdy \tag{4.2}$$

where $\nabla I = [I_x I_y]^t$ is the spatial derivatives of the image I in the x and y dimensions. I_t is the derivative in time. The equation is solved with Gauss-Seidel method using an iterative schema.

Additional constraints like gradient constancy and affine transformation can be posed to further refine the early approaches.

Because of the existence of some image sequences with ground truth flow fields, training and learning using those ground truth flow fields can dramatically improve the results. For example, using spatial statistics as in [23], and learning methods as in [16].

MRF methods such as graph cuts and belief propagation have also been introduced to optical flow computations due to their success in stereo matching. Such as in [7].

4.2 Our Approach

A similar approach as in our stereo matching algorithm is adapted and applied to optical flow computations.

Consider all of the pixels in image I as a point set $X = x_1, x_2, ..., x_N$, and make all of the possible disparities as a label set $L = 1, 2, ..., n$, where N is the total number of pixels, and n is total number of disparity labels.

An initial disparity likelihood matrix $F0$, for pixel $i \in X$, with a flow vector $d \in L$ is given by the inverse exponential of the Sum of Squared Distances (SSD)

$$F0(i,d) = exp(-\frac{\sum_{j \in N_i} w_{ij} |I(j) - I'(j+d)|^2}{2\sigma^2 \sum_{j \in N_i} w_{ij}}) \tag{4.3}$$

where SSD is computed in the neighborhood of point i including i itself, j are the neighbors of i, which could be 4-neighborhood, or k Nearest Neighbors (kNN). w_{ij} is the edge weight between i and j. This form works like an adaptive support,

which assigns strong weights on similar pixels with i in the local neighborhood. The inverse exponential converts the SSD into a similarity measure as well as regulates the disparity field F between 0 and 1. σ is a regularization factor. $F0$ is a matrix of size $N * n$ computed on all points in X, with every possible label $d \in L$.

The disparity likelihood matrix F^0 is then normalized into a soft assigned probability matrix

$$F^0(i,d) = \frac{F0(i,d)}{\sum_d F0(i,d)} \tag{4.4}$$

where each element $F^0(i,d)$ represents the initial probability of point i to be given a disparity vector d. F^0 is then a non-negative matrix and its row sum is 1.0. Intuitively, the bigger $F^0(i,d)$ of a given point i, the higher the probability that point i is assigned with disparity d.

As illustrated in Chapter 3, without further constraint, the MAP results using directly the initial flow cost volume F yields very noisy results.

4.2.1 Diffusion Process

A similarity matrix W is constructed by

$$W_{ij} = \begin{cases} exp(-\frac{|I_i - I_j|^2}{2\sigma^2}) & i,\ j \in N \\ 0 & otherwise \end{cases} \tag{4.5}$$

And the random walk matrix is given by

$$S = D^{-1} \cdot W \tag{4.6}$$

where D is the degree matrix of W

$$D_{ik} = \begin{cases} \sum_j W_{ij} & \text{if } i = k \\ 0 & \text{if } i \neq k \end{cases} \tag{4.7}$$

Similar to Chapter 3, the closed-form solution of the diffusion process is given by

$$F^* = (1 - \alpha S)^{-1} F^0 (1 - \alpha) \tag{4.8}$$

The maximum likelihood disparity field T using the cost volume F can be computed as

$$T(i) = \max_{d \in L} F(i, d) \tag{4.9}$$

The diffusion end result F, has taken care of the matching costs of neighboring pixels for every individual pixel. As shown in later sections, the resulting flow fields become much smoother and are improved significantly.

4.2.2 Noise and Outlier Removal

Noise exists in optical flow computations for various reasons. First of all, optic flow estimation itself is an ill-posed problem because it is seriously under constrained. Both the brightness constancy constraint and the piece-wise smoothness constraint are not necessarily correct in every circumstance. Among other reasons, the most significant ones are occlusions, ambiguities in homogeneous areas and illumination variations in different frames.

So first, we need to detect those noises and outliers. Then we need some algorithms to find out the real underlying flow fields of those outliers.

For the first problem, that of noise detection, a confidence measurement is introduced to reflect our confidence in whether the computed flow fields are correct or not. For the second problem, that of outliers fitting, the flow fields with high confidence will be deemed to be ground truth and outliers will be solved in a semi-supervised fashion.

Confidence Measurements

Currently, there are several clues we can use to detect outliers and noise. First, the normalized maximum likelihood T on each pixel is an important measurement of the confidence. The $N * n$ cost volume F is normalized so that the row sum is 1. The higher T, the higher our confidence that the current label is correct.

As in stereo matching, cross checking optical flow fields back and forth gives an important clue for errors. Due to the speed concern, it is not used in the current

framework.

Another clue is that pixels with labels quite different from their neighboring pixels (esp. those not at edges) can be an outlier.

Outliers Removal by Iterative Propagations

A small portion of the pixels are recognized as outliers using the confidence measures introduced above. Our next problem is how to use the labels of the "good" pixels, to induce the true label of those outliers.

For this purpose, an heuristic approach similar to the algorithm in Chapter 5 is introduced, and we will present it briefly before the formal introduction in next chapter.

The traditional approach of Markov Random Field is to minimize the sum energy E of a data term E_d and a smoothness term E_s

$$E(f) = \sum_p D_p(f_p) + \sum_{p,q \in N} V_{pq}(f_p, f_q) \tag{4.10}$$

Where f_p is the flow vector at node p, $D_p(f_p)$ is the data term,, and $V_{pq}(f_p, f_q)$ is the smoothness term, between neighboring nodes p and q. $E(f)$ is composed of the sum of all data terms and smoothness terms.

In our work, we decompose the total energy into energy at individual pixels.

$$E(f) = \sum_p E_p \tag{4.11}$$

where E_p is the energy at pixel p,

Within E_p, the data term $D_p f_p$ is associated with each pixel, and the smoothness term $V_{pq}(f_p, f_q)$ corresponds to the edge weight of two neighboring pixels, p and q. We split the smoothness term equally into two parts,

$$V_{pq}(f_p, f_q) = \frac{1}{2} V_{pq}(f_p, f_q) + \frac{1}{2} V_{qp}(f_p, f_q)$$

and let the first part solely belong to pixel p, and similarly, make the second part fully belong to the pixel q.

And now, E_p becomes,

$$E_p = D_p(f_p) + \frac{1}{2}\sum_{q \in N_p} V_{pq}(f_p, f_q) \tag{4.12}$$

Assume the labels are updated in a given sequence, and for the time being, l labels y_i $i = 1...l$, at $X_l := x_i$ $i = 1..l$ are known. The next label to choose should be selected so that E_p is minimized. A potential candidate x_{l+1} is chosen from all neighbors of the l known points X_l, of which we have some knowledge, due to smoothness constraint imposed on its neighborhood.

A possible problem is that, not all neighbors of a potential candidate x_{l+1} are known at this time, so only the current knowledge about this pixel is used, and the equation is changed slightly to:

$$E_p = D_p(f_p) + \frac{1}{2}\frac{\sum_{q \in KN_p} w_{pq} V_{pq}(f_p, f_q)}{\sum_{q \in KN_p} w_{pq}} \tag{4.13}$$

where KN_p represents the known neighbors of pixel p, and the smoothness terms between p and all its KNOWN neighbor are weighted by w_{pq}.

The beauty of this approach is that first only a small proportion of the labels need to change. Most labels have high confidence and are set to inlier, only those labels that have been classified as outliers need to be updated. Second, we can update the best labels dynamically considering both the data term and the smoothness term, without the need for multiple iterations for each pixel. The above two points make the dynamic algorithm used in optical flow very fast as well as makeing the result much better and smoother in the flow field. (The diffusion process makes the probability field smooth, and not the field of labels in all circumstances). The possible drawback of this approach is the same as with any heuristic algorithm, which depends on the order of operations, and is an approximation instead the exact solution. (So is graph cut and message passing algorithm).

4.2.3 From Discrete to Continuous Flow Vectors

To give a solution with sub-pixel accuracy, a quadratic polynomial interpolation method is employed.

The constructed cost volume is discrete, so the flow fields are discontinuous. Quadratic polynomial interpolation is used to approximate a continuous cost function in 2D, from three discrete depth values, d, d_- and d_+, at pixels x_0, $x_0 - 1$ and $x_0 + 1$ separately.

Assume the underlying quadratic function is $f(x) = ax^2 + bx + c$, with the maximum at $x_{min} = \frac{-b}{2a}$, for $a < 0$.

we have

$$\begin{aligned} d &= ax_0^2 + bx_0 + c \\ d_- &= a(x_0 - 1)^2 + b(x_0 - 1) + c \\ d_+ &= a(x_0 + 1)^2 + b(x_0 + 1) + c \end{aligned} \tag{4.14}$$

we can get,

$$\begin{aligned} a &= \frac{d_+ + d_- - 2d}{2} \\ b &= \frac{d_+ + d_- - 4ax_0}{2} \end{aligned}$$

hence,

$$x_{min} = \frac{-b}{2a} = x_0 - \frac{d_+ - d}{2(d_+ + d_- - 2d)}$$

4.3 Hierarchical Approach

Because of the possibly large motion between consecutive frames, the matrices F or B could cost a huge amount of memory and computations. To solve this problem, we use a hierarchical approach, dividing both the image and labels into different levels from coarse to fine.

A pyramid approach is applied from coarse to fine. The bottom level is the original image. Each level is one fourth the size in each dimension of a lower level. A global motion, which is estimated at the coarser level, is refined and corrected at the finer level. Successively finer flow fields get their F by interpolating the predecessors and matching in the local neighborhoods, which have more details now.

4.4 Results

Experimental results for the Middlebury evaluation image sequences [1] are shown in 4.2. There are 8 image sequences to evaluate, in which Army, Mequon, Schefflera and Wooden are real sequences captured with a color camera. Grove, Urban and Yosemite are synthetic sequences, among which, Grove and Urban are color image sequences, and Yosemite is a gray image sequence. And Teddy is stereo image pair (which have a different ground truth flow field from the pair in Stereo evaluation).

The numerical results is given in the web address below. Our average error in end points is 0.83 across all 8 image sequences and is ranked the 10th on the flow evaluation table in

http://vision.middlebury.edu/flow/eval/results-huang/results-e1.html.

4.5 Discussion

4.5.1 Label Smooth Vs. Flow Field Smooth

In the popular energy minimization framework, such as Graph Cut or Belief Propagation, the underlining constraint is that the final label of the motion vector field should be smooth. We further enrich the concept by making the similarity flow field associated with each label smooth. In many cases, it could be observed that the label associated with the best cost field at a given pixel could be wrong, instead the 2nd or 3rd best label are the same with the ground truth. Thus, updating the whole flow fields is more meaningful and contains more information than updating only the best labels.

4.5.2 Advantages of Our Approach

The advantages of this approach are two-fold. First, the smoothness term is implicit, and can be computed easily with sparse matrix operations. Second, it allows for the motion vector field discontinuity in a small neighborhood, if the relative cost is big enough. Thus, finer details can be kept without too much smoothing. On the other hand, in the homogeneous untextured regions, the original costs inside could be close

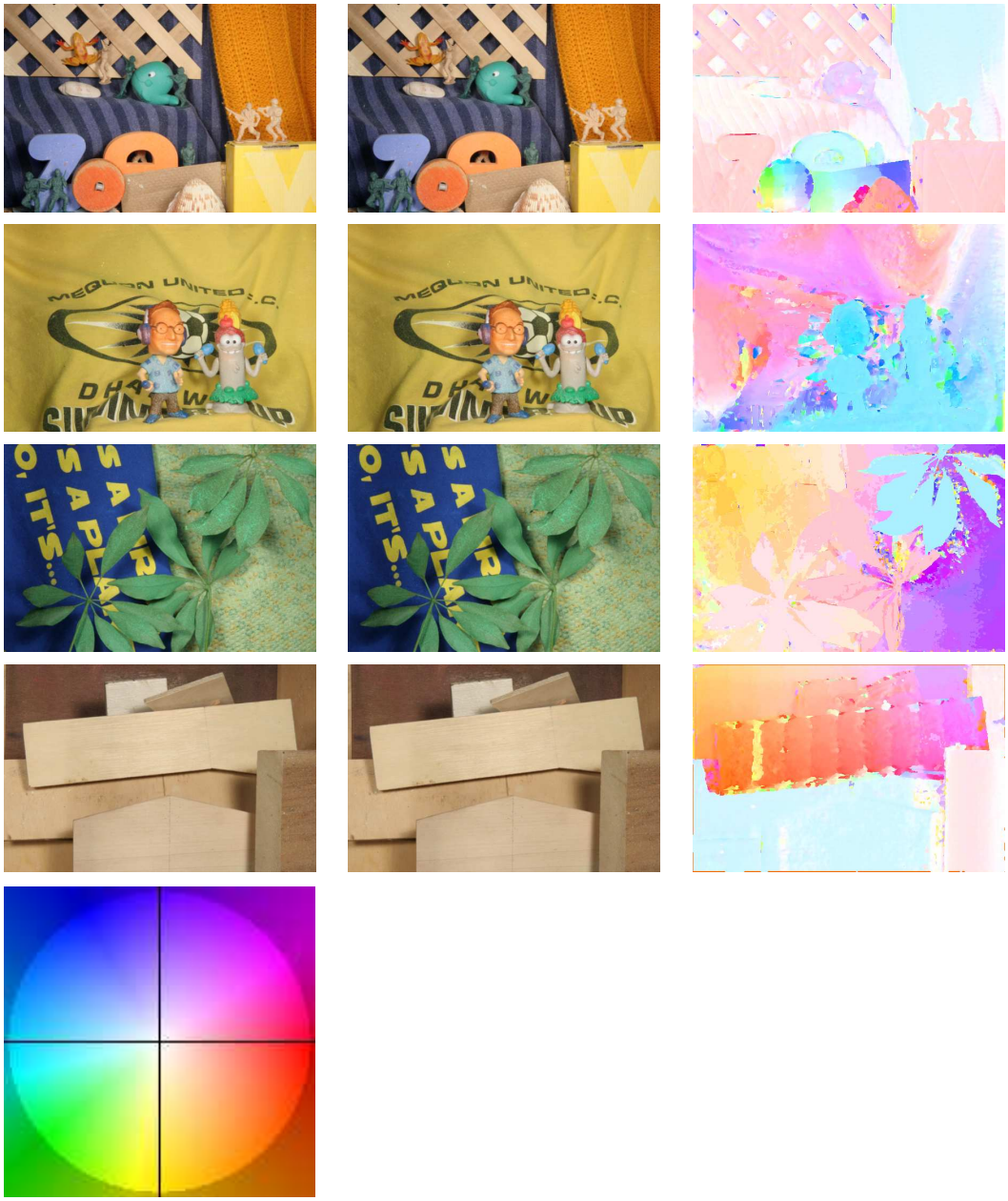

Figure 4.2: Results of Optical Flow sequences Army, Mequon, Schefflera and Wooden. From left to right, first frame, next frame, our result. From top to bottom, Army sequences, Mequon Sequences, Schefflera sequences and Wooden Sequences. Bottom, the color wheel, the flow encoding schema which color indicates flow orientation and saturations represent flow magnitudes.

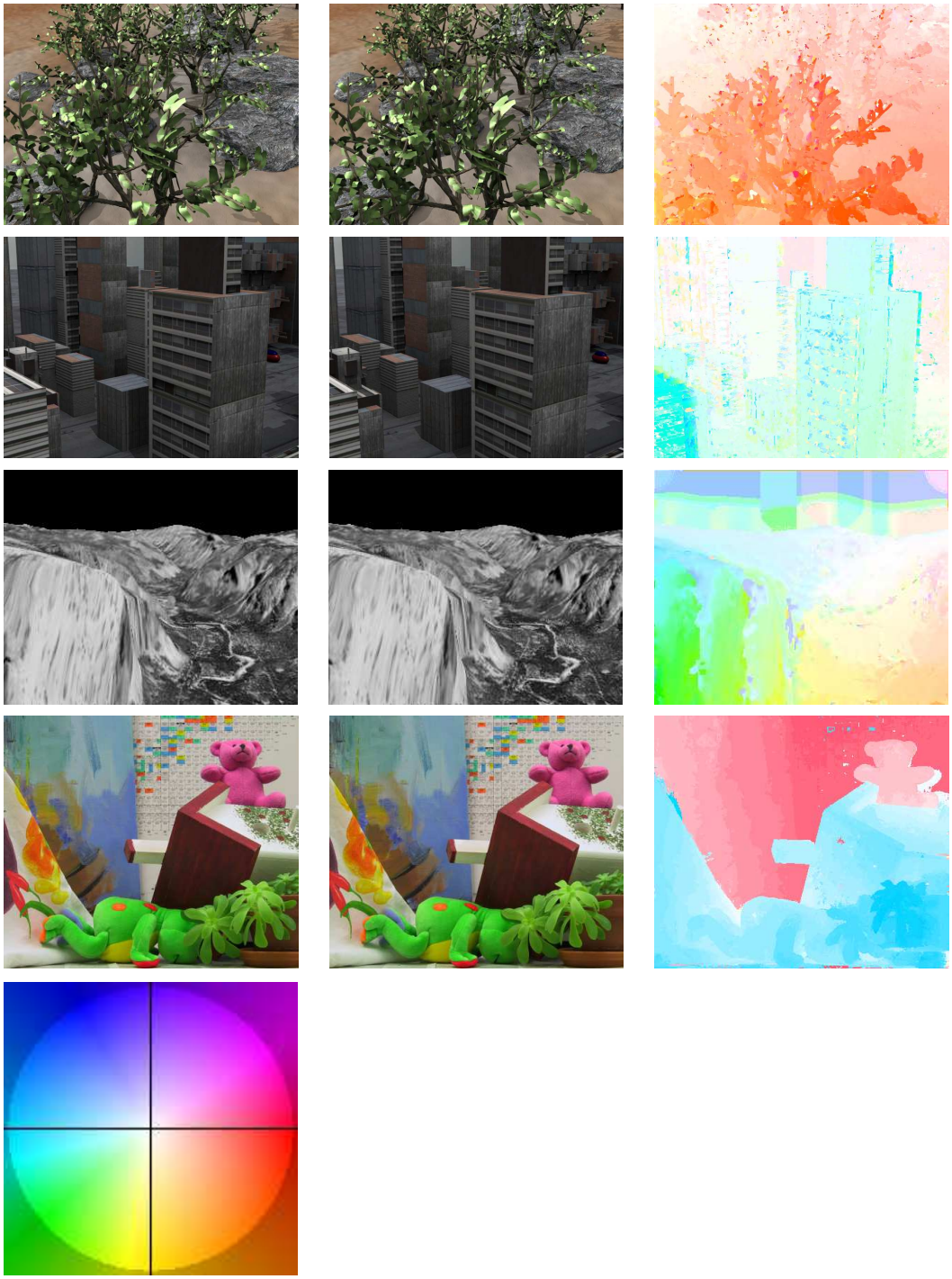

Figure 4.3: Results of Optical Flow sequences Grove, Urban, Yosemite and Teddy. From left to right, first frame, next frame, our result. From top to bottom, Grove sequences, Urban Sequences, Yosemite sequences and Teddy Sequences. Bottom, the color wheel, the flow encoding schema which color indicates flow orientation and saturations represent flow magnitudes.

for each label, distinctive points only reside at feature points or boundaries. But the higher similarity measure could be propagated to the similarity field to the relevant labels, and cause the cost field at other regions to be higher as well.

Using the confidence level of the matching scores, the scores of high confidence data are transductively propagated to low confidence ones. The best part of that is, no explicit learning is needed, and interactions like human labeling are not necessary. Thus it could be called unsupervised learning in a semi-supervised learning framework.

Chapter 5

Tracking with Semisupervised Learning and Sparse Feature Matching

The approaches in Chapter 3 and 4 are suitable for dense matching problems, such as stereo matching and dense optic flow estimation. They are substantially faster than existing graph cut and belief propagation algorithms. For problems like object tracking in image sequences, speed is an important factor and sometimes a sparse set of pixels is sufficient and can give reasonable results.

In this chapter, we combine sparse feature matching with dense optic flow, and a heuristic algorithm is proposed for computing optic flow, and object tracking in video sequences. The method is both computationally effective and fast. It is based on a good initialization by feature points matching, such as SIFT [17]. The matching is then propagated to neighbors of the initial points, which have the lowest energy, maintained by a Heap and extracted one at a time. Results are presented in optic flow, tracking and video object segmentation in video sequences.

The key observation of this work is that good initialization could ease the problem dramatically. Once some anchor points are fixed, all their neighbors should keep smooth with the anchors, as well as maintain their energy low. The *consistent* neighbors with the lowest energy will be added into the anchor points, and the process will continue until all points are added. The anchor points could be initialized by feature point detection and matching.

Both initialization and the order of propagation can introduce errors into the final result. Upon completion of propagation, the final result is used to update earlier points which do not have enough knowledge at that time, and this is carried out using harmonic functions [33].

In joint segmentation and tracking, the object of interest will be represented by a compositional model of different segments. Shape priors for each segment will be kept, updated and warped in each new frame, and combined with the optic flow field to make the results accurate and robust.

5.1 Related Work

5.1.1 Feature Tracking

Scale-invariant feature transform (SIFT) [17] is a scale-space extrema detection and description algorithm, and because of its invariance to scale and rotation, it is a very robust method to find the keypoints, the feature points on an image. Other methods includes the famous Kanade-Lucas-Tomasi tracker (KLT) [19] [28], and GLOH [20], which uses a spatial histogram scheme. A good review of feature descriptors could be found in [21]. SIFT is used in this work, but any other method could be used interchangeably.

Zhu et al. [33] gives an approach for the semi-supervised learning or classification problem, by relaxing the discrete label field into a continuous one, and a point is labeled as the nearest labeled example in a random walk sense. A similar approach has been used in random walk, an interactive image segmentation technique by Grady [9]. The details can be found in Chapter One of this dissertation.

5.1.2 Video Object Tracking and Segmentation

Video object cutout by Wang et al. [29] is an interactive video segmentation method. First, the whole video sequence is hierarchically decomposed using mean-shift. The user has to provide labels in different frames and locations, then the video object is segmented by min-cut optimization. This is very useful in post processing of video, but not in real time when not all frames are known in advance. Zitnick et al. [34] proposed a method for computing optical flow by finding the consistent segments in video using mean shift. The final result of optic flow and segmentation is on the segment level, depending on over-segmentation by mean shift, and is less accurate on the pixel level.

The updating procedures of the algorithm is also similar to the framework of fast marching [25] or Dijkstra's method [13]. Both have some Starting points, and iteratively add Neighbors with the minimum values into the Starting points.

5.2 Algorithm

Assume we are given two input images I and I', which are two consecutive frames in optic flow or tracking. We are trying to find Optic flow or disparity field at all nodes on image I, by energy minimization.

5.2.1 Energy Terms

The total energy to minimize in MRF is as follows:

$$E(f) = \sum_{p} D_p(f_p) + \alpha \sum_{p,q \in N} V_{pq}(f_p, f_q) \tag{5.1}$$

Where f_p is the flow vector at node p, $D_p(f_p)$ is the data term,, and $V_{pq}(f_p, f_q)$ is the smoothness term, between neighboring nodes p and q. $E(f)$ is composed of the sum of all data terms and smoothness terms, with α as a mixing factor for the Data term D_p and the Smoothness term V_{pq}.

In our work, we decompose the total energy into energy in individual nodes.

$$E(f) = \sum_{p} E_p \tag{5.2}$$

and

$$E_p = D_p(f_p) + \frac{1}{2}\alpha \sum_{q \in N_p} V_{pq}(f_p, f_q) \tag{5.3}$$

(without loss of generality, $\frac{1}{2}$ before $\alpha \sum_{q \in N_p} V_{pq}(f_p, f_q)$ could be omitted)

Intensity and gradient constancy are assumed for data term $D_p(f_p)$

$$D_p(f_p) = |I_p - I'_{p+f_p}| + |G_p - G'_{p+f_p}| \tag{5.4}$$

which is defined as the absolute difference between intensity I_p and the shifted intensity I'_{p+f_p} in the other image, plus absolute gradient difference between G_p and G'_{p+f_p}.

The simplest form of $V_{pq}(f_p, f_q)$ between neighboring flow vectors f_p and f_q is,

$$V_{pq}(f_p, f_q) = min(|f_p - f_q|, k) \tag{5.5}$$

for some constant k

Note label smoothness should be kept in homogeneous regions, but not necessarily at edges, where discrepancies are likely to happen. So neighboring pixels with similar intensities or dramatically different intensities should be assigned different smoothness terms, even if the flow vectors f_p and f_q are the same.

For this purpose, we choose the normalized edge weight function p_{pq} between p and q to weight the smoothness term. p_{pq} also defines the probability that flow f_q from node q will propagate to node p

$$p_{pq} = \frac{w_{pq}}{\sum_{q \in N_p} W_{pq}} \tag{5.6}$$

where edge weight

$$w_{pq} = exp(-\frac{|I_p - I_q|^2}{2\sigma^2}) \tag{5.7}$$

and σ is user defined.

This term suggests that the smoothness term between neighboring pixels depends on the similarity of their intensities. This ensures that pixels need to be smooth with only neighbors of similar intensities. For neighboring pixels across edges, large difference in flow vectors is allowed. This is also in accordance with the random walk sense, where labels propagate into neighbors according to the probability of label propagation.

So finally, the smoothness term becomes

$$V_{pq}(f_p, f_q) = min(|f_p - f_q|, K) * P_{pq} \tag{5.8}$$

5.2.2 Initialization

First, feature detection is applied and matches are found in both input images. The flow vectors of all matches will be sorted and those with a large standard deviation will be discarded as outliers. The range of flow will be determined by the matches and the flow is computed for the feature points.

There are three types of labels marking the status of every node, *KNOWN*, *NEIGHBOR* and *FARAWAY*. Nodes of *KNOWN* are ones in which their labels f_p are known so far. *NEIGHBOR* nodes are neighbors of *KNOWN*, the rest of the nodes are marked with *FARAWAY*. A heap structure is used to hold *NEIGHBOR* nodes, to increase the speed of sorting in finding the nodes with the minimum value.

Those detected feature points are labeled as initial points, and are added into *KNOWN* We will assume it is accurate for the time being, and it will be refined in post-processing steps. The neighbors of the initial points are marked as *NEIGHBOR*, and added into a Heap. All other pixels are initialized as *FARAWAY*.

5.2.3 Iteration of Propagations

The basic iteration steps are as follows

[1] Select a node p from *NEIGHBOR*, which has the lowest energy E_p

[2] Add p to *KNOWN*

[3] **a.** Add all p's neighbors in *FARAWAY* to *NEIGHBOR*, compute their energy

b. Update p's neighbors which are already in *NEIGHBOR*

[4] return to step [1] until no points are left in *NEIGHBOR*

If the flow fields of all neighbors of point p are known already, f_p could be decided deterministically. In many cases, we only know some of its neighbor's flow field f. We would like a node of which we have a better *knowledge* as well as with lower energy, to be selected first to minimize the overall error.

An easier way to define our *knowledge* of a node p is to count the current known neighbors of p. The more neighbors in *KNOWN* it has, the better we can determine its label l_p.

For the same reason we mentioned in the previous section, the normalized edge weight funtion w_{pq} in equation (5.7) has to be considered to take neighborhood node similarities into account. The normalized sum of edge weights between p and all its *known* neighbors is used to define our knowledge of p,

$$K_p = \frac{\sum_{q \in KN_p} w_{pq} V_{pq}(f_p, f_q)}{\sum_{q \in KN_p} w_{pq}} \tag{5.9}$$

So for the time being, the energy at a neighboring node p is defined as

$$E_p = D_p(f_p) + K_p \tag{5.10}$$

which is composed of a data term $D_p(f_p)$ at p, plus the weighted sum of the smoothness terms to its *known* neighbors. (Since only neighbors of p in *KNOWN* is available when p is being accessed), and the *knowledge* term K_p in (5.9). The *knowledge* term also makes the boundaries of *KNOWN* points smooth, as the propagation tends to first go over all familiar regions before marching into unknown territories. So this also makes an effective curvature term.

The general principle of Propagation is to choose the best easiest node for label assignment with the lowest energy E_p first, and leave the harder nodes with the highest energy last, to minimize the overall energy E. The nodes with lowest energy and better *knowledge* tend to have lower errors, since they have a combined lower data term, which indicates they have both a good fit and a lower smoothness term, which means they are similar to their neighbors.

In iteration step[3], if q is in *FARAWAY*, its energy E_q will be generated as in equation (5.10). If q is already in *neighbor*, we will update its energy E_q , since p is just added to *known*, the smoothness term will have additional w_{pq} normalized $V_{pq}(f_p, f_q)$, as in (5.9) as well. The energy is updated for all possible f_q.

The iterations will continue until there are no more points in the neighbor Heap.

5.2.4 Post-Processing

The iteration is heuristic, and the final result could be determined by the initialization and updating order, so postprocessing is necessary. A harmonic function is employed for this purpose.

A harmonic function f satisfies

$$f(x_i) = y_i \tag{5.11}$$

And it minimize

$$\sum_{ij} w_{ij}(f_i - f_j)^2 \tag{5.12}$$

where ij are all pairs of i and j that are neighbors, and the average of neighbors i is

$$f_i = \frac{\sum_{j \in N_i} w_{ij} f_j}{\sum_{j \in N_i} w_{ij}} \tag{5.13}$$

where the sum is over all j which are neighbors of i. If some prior labels are known and fixed, there exists a closed form solution for harmonic function. The harmonic function could also be interpreted as random walk and is related to the graph Laplacian matrix.

The x and y field of the flow vector will be extracted and each will be averaged with the harmonic function for a few iterations, to ensure spatial consistency of labels.

5.3 Combined Tracking and Segmentation

In this section, we will show that the above algorithm could be adapted and applied to accurately track and segment an object simultaneously in real time.

5.3.1 Iterative Segmentation and Optic Flow Computation

Assume an initial mask of the object of interest is either given or computed previously (for example, using optic flow). In this case, in addition to the disparity or flow vector f_p, segment labels l^i are introduced. l^i could be set to binary label $i = 0, 1$, where l^1 are foreground pixels, and l^0 are background pixels. Or we could make a compositional model of M different parts, where $i = 0, 1...M$. $i = 0$ stands for background, and $i = 1...M$ stands for different parts of the object or different objects. The compositional model could be retrieved with a Gaussian Mixture Model or Mean Shift [5].

The introduction of the segment label l solves the segmentation problem in conjunction with the optical flow field. We will show that both the flow vector f and segment label l could be processed properly using the iterative propagation framework in Section 2.

The energy model has been modified slightly, to incorporate energy from segmentation and the smoothness of segment labels between neighbors,

$$E(f) = \sum_{p} D_p(f_p) + \sum_{q \in N_p} V_{pq}(f_p, f_q) + \sum_{p} P(l'_{p+f_p} - l_p) + \sum_{q \in N_p} U_{pq}(l_{p+f_p}, l_{q+f_p}) \quad (5.14)$$

For the last two terms, $P(l'_{p+f_p} - l_p)$ accounts for the segment label at $p + f_p$ in the next frame being mapped to the same segment at p in previous frame, given the flow vector f_p. l_p is the label in the previous frame, f_p is the flow vector, and l'_{p+f_p} is label of p in next frame. We will impose a penalty term $P(l'_{p+f_p} - l_p)$, if p is mapped to another segment after the optic flow between consecutive frames.

$U_{pq}(l_{p+f_p}, l_{q+f_p})$ is the smoothness term for the neighboring nodes which remain in the same segment on the new frame.

The same iterative schema in Section 3 will be used to update the segment label and optic flow field simultaneously.

5.3.2 Postprocessing

Instead of using the labels l_p for segmentation directly, information from previous frames will also be considered. Knowledge of all previous frames is giving us important prior shape information for each part and the whole object being tracked. The flow field f_p could determine a rigid or non-rigid transform Φ^i for each segment i. Previous shape priors could be warped into the new frame given the individual transform of each part. The warped shape prior and the label computed in the last section could be combined by a weighted sum, to give the probability at the current frame.

Since this is a multi-label segmentation problem, a probability matrix Ψ^i_p will be given to describe the likelihood for p to be assigned with segment label i. Where Ψ^i will be 1 at the maximum of the sign distance function of l^i, and 0 when far away from label l^i.

$$\hat{\Psi^i_p} = \alpha T^i(\Phi^i_p) + (1 - \alpha)\Psi^i_p \quad (5.15)$$

Where α is the learning factor, Φ^i_p is the previous shape prior for segment part i, T^i is the transform of component i, and $\hat{\Psi^i_p}$ is the new estimation.

The probability matrix Ψ^i will be updated at the end after probability propagation, using the past history and result in this frame, again weighted by α.

5.3.3 Probability Propagation

Now instead of optic flow field, we are now more interested in the probability function of different segments. So the probability function of each segment is individually averaged by the harmonic function (5.13). The final label will be taken as the maximum across all possible labels.

5.4 Result

In this section, the result of stereo, optic flow and combined tracking and segmentation will be shown.

Figure 5.1 shows an example of stereo matching for illustration purpose. The top row shows the original stereo pair of Teddy and the ground truth of disparity. The initial points as well as some intermediate iterations steps have been shown to illustrate the iteration process. Notice in iter. 100000 (bottom left), the propagation seems to stops at image edges, because all edge points have larger energy term E_p than homogeneous regions.

The final iteration is shown in the bottom middle, the disparity result looks fairly good. The result after post-processing is shown in bottom right. The image size is $450 * 375$, and the disparity field is large, from 0 to 60. Yet, the computation takes only a few seconds, e.g. 1.2 seconds for SIFT feature matching, 2.2 seconds for the iteration process, and 0.17 second for the post-processing (10 iterations). This result is comparable with other MRF methods, but takes significantly less computation time.

Another example is shown for optical flow in Figure 5.2. The optic flow in all frames shows great accuracy and the boundary of the moving human body and back of trunk could be clearly seen. The image size is $640 * 480$, with 7 frames in total. The starting and ending frame as well as every other frame of the optic flow are shown in Figure 5.2.

The next example Figure 5.3 demonstrates the results for combined segmentation

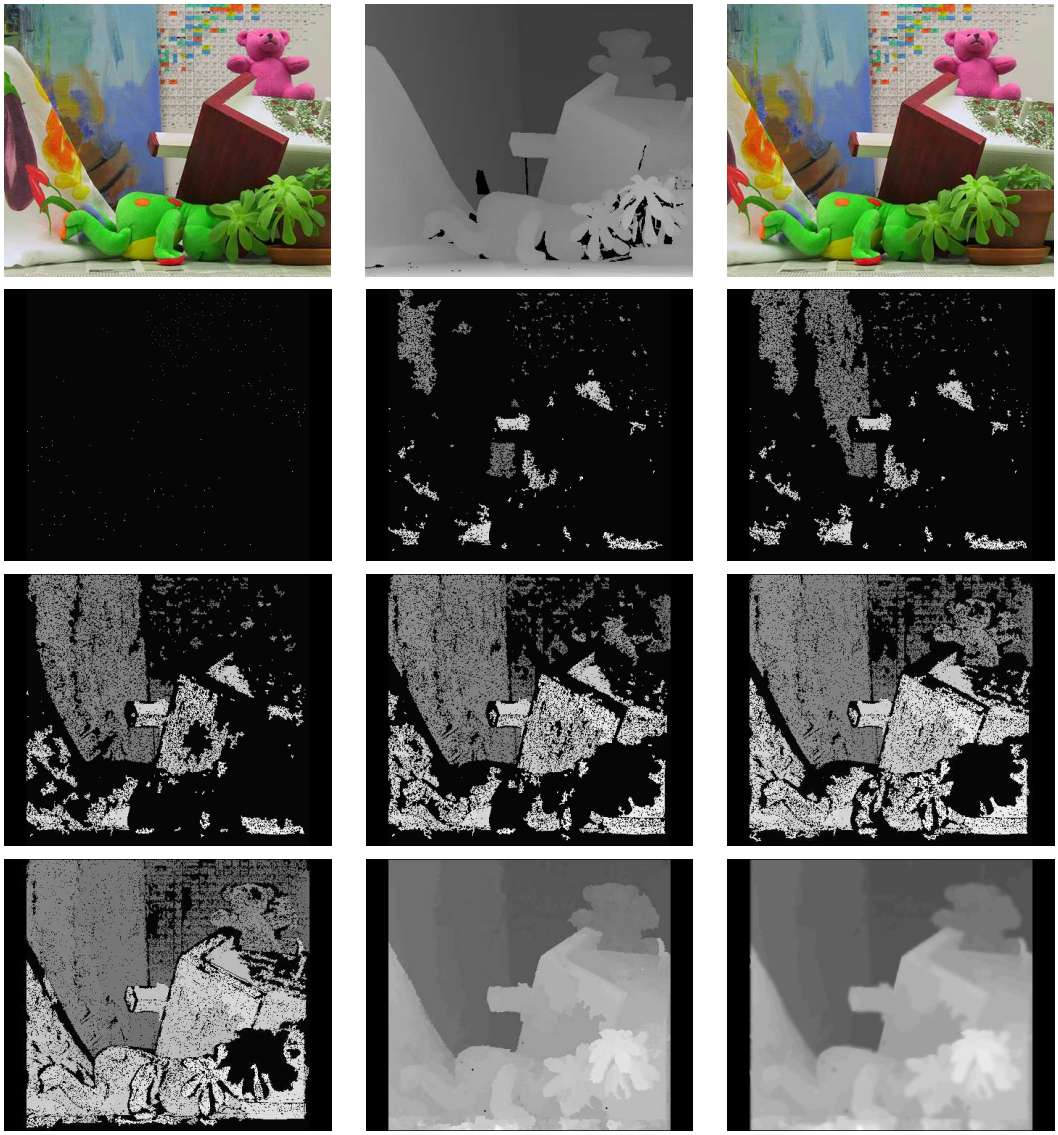

Figure 5.1: Results of stereo of Teddy. 1st row: from left to right, left image, ground truth, right image. 2nd row: initial points, iter 10000 , iter 20000. 3rd row: iter 40000, iter 60000, iter 80000. 4th row: iter 100000, result after final iter, result after postprocessing

Figure 5.2: Results of Optic flow of MiniCooper sequence [1]. from left to right, top to bottom: first frame, last frame (8th). optic flow field in frames 1, 3, 5, 7 (only X field are shown)

and tracking a tropical fish with a moving camera. The video size is 320*240, with 190 frames and the fish undergoes non-rigid transform as it changes its direction in the water. The red lines on top the original image marks the boundarys of the fish as well as different composing parts. The binary gray in the middle column shows the sum probability to be the object, and the gray image on the right column shows the segmentation label of each part. An roughly initial mask is given for frame 1 (not shown here). The SIFT matching is only computed inside the fish to save computation. Mean shift is applied to compute initial segments in the first frame only. The probability matrix for each segment are updated as the tracking moves on, and mapped to the new frame using affine warping.

5.4.1 Discussion

The computation of the Heap is $O(NlogN)$, where N is the number of pixels, $logN$ comes from the heap operation, insertion and extractMin. It is much smaller in practice, since not only a portion of the N nodes are in the heap at the same time. Postprocessing will take $O(N)$ time, and could be negligible. $O(NlogN)$ is already much inexpensive in computation than the polynominal computation time of graph cut.

In the accurate object tracking and segmentation case, only a small area inside and

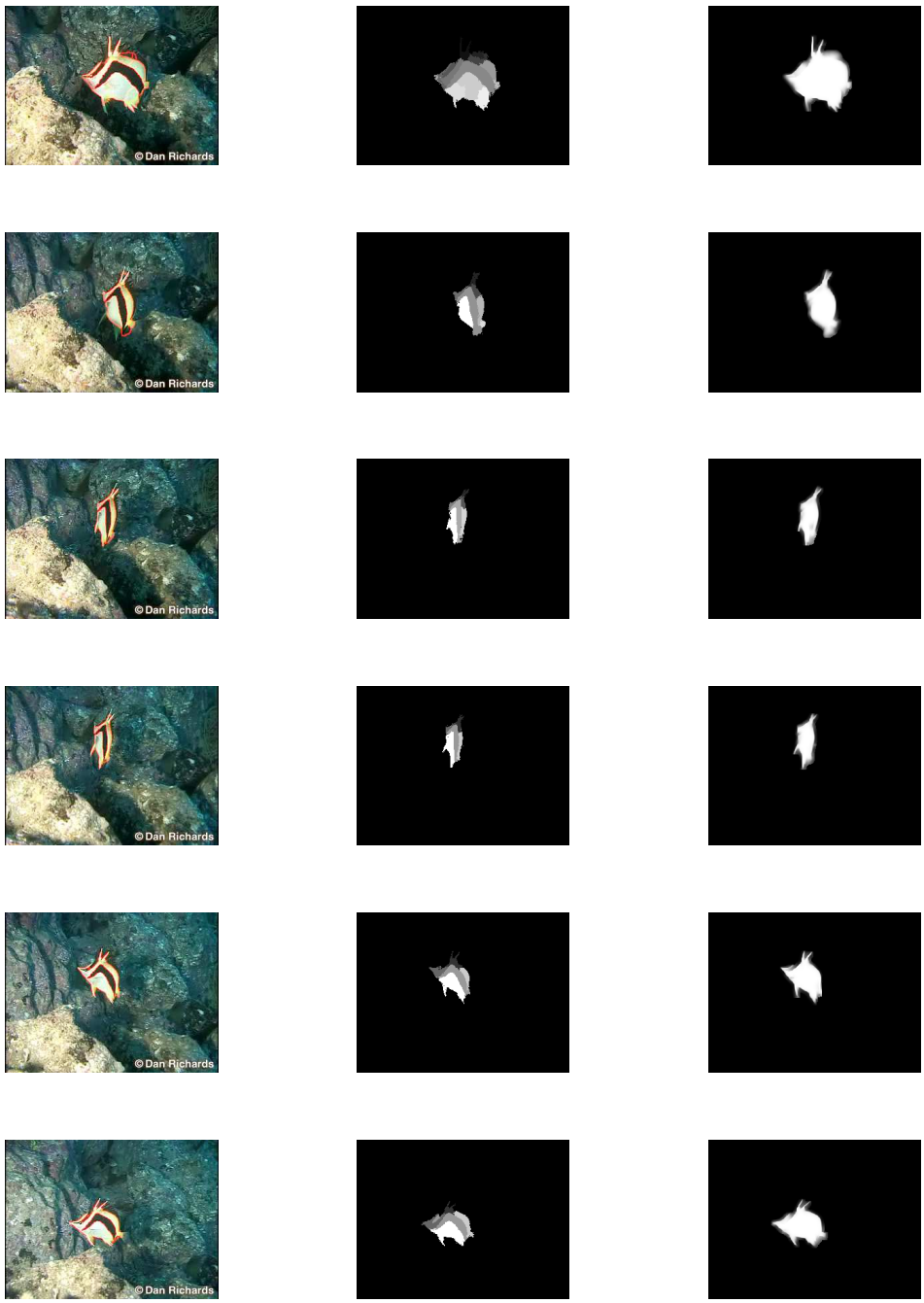

Figure 5.3: Results of Combined segmentation and tracking of a fish. The red lines on top the original image marks the boundary of the fish as well as different composing parts. The gray image on the middle column shows the segmentation label of each part and the gray image in the middle column shows the sum probability of the object. The 1st through 6th row show frame 25, 45, 65, 85, 105, 125 in the video sequence.

around the object of interest is needed for computation. Thus only 0.23 secord is needed per frame in average. And Near-real time tracking could be achieved.

Our approach requires little or none user interaction for segmentation of video, and achieves accurate and reliable results.

Chapter 6

Discussion and Conclusions

6.1 Discussion

6.1.1 Computational Complexity

In the diffusion equation for stereo or optical flow,

$$F^{t+1} = \alpha S F^t + (1-\alpha) F^0$$

the random walk matrix S is a sparse $N*N$ matrix, which only involves the neighbors of a given point (4-neighborhood is used), thus each iteration is of $O(N)$ in computation.

For the closed form solution,

$$F^* = (1-\alpha S)^{-1} F^0 (1-\alpha)$$

it involves the inverse of the sparse matrix S. At first, it looks daunting. But in Matlab, Backslash or matrix left division are used to solve equation $Ax = B$, where A is a square matrix, B is a matrix with several column vectors. Usually it is computed by Gaussian elimination where computation is $O(n^3)$ in general, but for sparse matrix, especially tridiagonal matrix and penta-diagonal matrix, algorithms with $O(N)$ computations exists [8] [12]. The sparse random walk matrix S is not pentadiagonal, but it could be rearranged into a pentadiagonal-like matrix. In reality, the computation also increases linearly with the number of labels n, where $n << N$.

The computation of our algorithm is in general $O(N)$ instead of the polynomial approach in Graph cut and Message Passing algorithms. The most recent logCut algorithm [15] is still several times slower than our approach even though it is supposedly an $O(NlogN)$ algorithm.

There are additional overheads for the post-processing of stereo matching and optical flow, but the computational time is less than or of the same order as the closed-form diffusion process.

Our computation time on the Tsukuba pair in stereo, with an image size $352 * 288$ and labels ranging from 0 to 15, takes about 2 min using Matlab. A less accurate implementation of the same algorithm using C++ only takes 4 seconds. Our algorithm is much faster than other algorithms which have posted their computational time.

For optical flow, the computation time of one of the most computationally intensive image pairs, Urban, takes about 3 min using Matlab with C++ mex function. The image is bigger, with a size of $640 * 480$, and has a much bigger range of possible flow fields, with approximately an X flow field from -32 to 3, and a Y flow field from -1 to 9, and thus a label range of 432. The optical flow is faster in general than stereo since there will be no cross-checking and we do not make the same assumption that each segment lies on a piece-wise smooth plane.

6.2 Conclusions

A novel algorithm to solve the general image matching problem using graph theoretic semi-supervised learning is presented in this dissertation. The algorithms have been adapted and applied in different applications, such as stereo matching, optical flow and object tracking. State of art results using this approach have been demonstrated in all of these research areas.

The main contribution of this dissertation is to apply graph based semi-supervised learning in stereo, optic flow and tracking, in a manner which has not been proposed before so far as the author knows. In fact, the supervised labels are obtained automatically using high confidence points (as in Stereo or Optic Flow), or a sparse feature matching method (like SIFT). Hence, no human interaction is needed for the matching process, and it is essentially an unsupervised method. Only in terms of theory, is it based on the semi-supervised learning algorithm.

Graph based Semi-supervised learning has been used in interactive image segmentation, as in random walk [9], and more recently, in object recognition, [14]. The scope has been broadened to dense image matching in this work.

The results show the advantages of our approach, it is robust, accurate and computationally efficient. In comparison with the state of art, in the standard Middlebury database evaluation of stereo and optical flow, our results are comparable in accuracy with the top algorithms, and are generally more computationally efficient.

6.3 Future Work

We integrated sparse feature matching and dense image matching using semi-supervised learning, to some extent, in Chapter 4, the optical flow part of this work. But there is still a lot more to exploit in this area.

To further improve the numerical accuracy of our algorithms, we can dynamically update the weights matrix and make it more discriminative for different labels. The fixed edge weight matrix is one of the bottlenecks where we could not improve at a certain point.

This framework could also be easily transfered to medical image registration or tracking. For example, registration across MR and CT images. A very suitable area in tracking is on ultra-sound heart image sequences, where the chamber could be tracked across frames. An underlying transformation, affine or even non-rigid, could be assumed between source image and target image to register. The transformation will be incorporated implicitly in the semi-supervised graph diffusion framework using our current approach.

The other area is unsupervised image segmentation. The graph cut and random walk algorithms both need user interactions, and user-provided seeds are used. In a semi-supervised diffusion framework, we could ease the necessities of user interactions by semi-supervised clustering of the distance of pixels, and update the edge weight dynamically.

For the continuation of our NSF funded project, Real time Identification and Monitoring for Reef Fish Communities, the color and shape model inside the fish contour will be retrieved and compared with previous stored models, and recognized as the fish with the most similar pattern. If this pattern has not appeared before, it will be saved as a new model. A database for different types of fish, at difference poses or illuminations will be stored and sorted in a database.

References

[1] S. Baker, S. Roth, D. Scharstein, M. Black, J. P. Lewis, and R. Szeliski. A database and evaluation methodology for optical flow. Oct. 2007.

[2] R. C. Bolles and M. A. Fischler. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. In *Image Understanding Workshop*, pages 71–88, 1980.

[3] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. 23(11):1222–1239, November 2001.

[4] O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.

[5] D. Comaniciu and P. Meer. Mean shift analysis and applications. In *International Conference on Computer Vision*, pages 1197–1203, 1999.

[6] O. Duchenne, J. Y. Audibert, R. Keriven, J. Ponce, and F. Segonne. Segmentation by transduction. In *IEEE Computer Vision and Pattern Recognition or CVPR*, pages 1–8, 2008.

[7] B. Glocker, N. Paragios, N. Komodakis, G. Tziritas, and N. Navab. Optical flow estimation with uncertainties through dynamic MRFs. In *IEEE Computer Vision and Pattern Recognition or CVPR*, pages 1–8, 2008.

[8] G. H. Golub and C. F. V. Loan. *Matrix computations*. John Hopkins, Baltimore MD, 1983.

[9] L. Grady. Multilabel random walker image segmentation using prior models. pages I: 763–770, 2005.

[10] L. Grady and G. Funka-Lea. Multi-label image segmentation for medical applications based on graph-theoretic electrical potentials. In M. Šonka, I. A. Kakadiaris, and J. Kybic, editors, *Computer Vision and Mathematical Methods in Medical and Biomedical Image Analysis, ECCV 2004 Workshops CVAMIA and MMBIA*, number LNCS3117 in Lecture Notes in Computer Science, pages 230–245, Prague, Czech Republic, May 2004. Springer.

[11] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artif. Intell.*, 17(1-3):185–203, 1981.

[12] A. A. Karawia. A computational algorithm for solving periodic penta-diagonal linear systems. *Applied Mathematics and Computation*, 174(1):613–618, 2006.

[13] Knuth. A generalization of dijkstra's algorithm. *IPL: Information Processing Letters*, 6, 1978.

[14] C. Leistner, H. Grabner, and H. Bischof. Semi-supervised boosting using visual similarity learning. In *IEEE Computer Vision and Pattern Recognition or CVPR*, pages 1–8, 2008.

[15] V. Lempitsky, C. Rother, and A. Blake. Logcut: Efficient graph cut optimization for markov random fields. In *International Conference on Computer Vision*, pages 1–8, 2007.

[16] Y. P. Li and D. P. Huttenlocher. Learning for stereo vision using the structured support vector machine. In *IEEE Computer Vision and Pattern Recognition or CVPR*, pages 1–8, 2008.

[17] D. G. Lowe. Object recognition from local scale-invariant features. pages 1150–1157, 1999.

[18] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. DARPA Image Understanding Workshop*, pages 121–130, 1981.

[19] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. pages 121–130, 1981.

[20] K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. pages 525–531, 2001.

[21] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, Oct. 2005.

[22] A. Y. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

[23] S. Roth and M. J. Black. On the spatial statistics of optical flow. 2007.

[24] D. Scharstein and R. S. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1-3):7–42, Apr. 2002.

[25] J. A. Sethian. *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 1999.

[26] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Conf. Computer Vision and Pattern Recognition*, June 1997.

[27] D. A. Tolliver and G. L. Miller. Graph partitioning by spectral rounding: Applications in image segmentation and clustering. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1053–1060, Washington, DC, USA, 2006. IEEE Computer Society.

[28] C. Tomasi and T. Kanade. Shape and motion from image streams: A factorization method part 3 - detection and tracking of point features. page required, 1991.

[29] J. Wang, P. Bhat, R. A. Colburn, M. Agrawala, and M. F. Cohen. Interactive video cutout. *ACM Transactions on Graphics*, 24(3):585–594, July 2005.

[30] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency, 2003. In 18th Annual Conf. on Neural Information Processing Systems.

[31] X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005.

[32] X. Zhu. *Semi-supervised learning with graphs*. PhD thesis, Pittsburgh, PA, USA, 2005. Chair-John Lafferty and Chair-Ronald Rosenfeld.

[33] X. Zhu, J. Lafferty, and Z. Ghahramani. Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. 2003.

[34] C. L. Zitnick, N. Jojic, and S. B. Kang. Consistent segmentation for optical flow estimation. In *International Conference on Computer Vision*, pages II: 1308–1315, 2005.

Vita

Ning Huang

2009	Ph. D. in Biomedical Engineering, Rutgers University
2000-2004	M. S. in Biomedical Engineering, Rutgers University
1995-2000	B. S. in Biomedical Engineering, Tsinghua University, Beijing, China
1995	Graduated from Harbin No. 3 High School, Harbin, HeiLongJiang Province, China
2001-2005	Graduate Assistant, CAIP Center, Rutgers University